Question2

Solution :

1. Setup

Let us denote $dp(i,j)$ is the smallest number of times moves from lower elevation to higher elevation along such a path from $(1,R)$ to (i,j), $and\ h(i,j)$ is the elevation of the terrain at (i,j).

2. Subproblems

Because we can only go down or go right, hence $dp(i,j)$ is only related to $dp(i-1,j)$ and $dp(i,j-1)$. So, the subproblem What's the number of moves from lower elevation to higher elevation along such paths from $(1,R)\ to\ (i-1,j)\ and\ (i,j-1)$ separately are as small as possible'. Hence, the subproblem is computing $dp(i-1,j)\ and\ dp(i,j-1)$. If we have solved the subproblem:

If $h(i-1,j) < h(i,j)\ and\ h(i,j-1) < h(i,j)$:Both direction will increase the time of climbs, so $dp(i,j) = \min\{dp(i-1,j), dp(i,j-1)\} + 1$.

If $h(i-1,j) > h(i,j)\ and\ h(i,j-1) > h(i,j)$:Both direction will not increase the time of climbs, so $dp(i,j) = \min\{dp(i-1,j), dp(i,j-1)\}$.

If $h(i-1,j) < h(i,j)\ but\ h(i,j-1) > h(i,j)$:Only go down will increase the time of climbs, so $dp(i,j) = \min\{dp(i-1,j)+1, dp(i,j-1)\}$.

If $h(i-1,j) > h(i,j)\ but\ h(i,j-1) < h(i,j)$:Only go right will increase the time of climbs, so $dp(i,j) = \min\{dp(i-1,j), dp(i,j-1)+1\}$.

3. Build-up order

Solve the subproblems in the order $dp(1,R), dp(2,R), dp(1,R-1), dp(2,R-1), dp(2,R-2)\ldots, dp(C,1)$.

4. Recursion

$dp(i,j)$

$$= \begin{cases} \min\{ dp(i-1,j), dp(i,j-1)\} + 1; if\ h(i-1,j) < h(i,j)\ and\ h(i,j-1) < h(i,j) \\ \min\{ dp(i-1,j), dp(i,j-1)\}; if h(i-1,j) > h(i,j)\ and\ h(i,j-1) > h(i,j) \\ \min\{ dp(i-1,j) + 1, dp(i,j-1)\}; if\ h(i-1,j) < h(i,j)\ but\ h(i,j-1) > h(i,j) \\ \min\{ dp(i-1,j), dp(i,j-1) + 1\}; if\ h(i-1,j) > h(i,j)\ but\ h(i,j-1) < h(i,j) \end{cases}$$

5. Base case

$$dp(1,R) = 0$$

6. Final solution

$$The\ number\ of\ moves = dp(C,1)$$

We can solve this problem by filling this table from top left to right bottom, the answer is in the blanket $dp(C,1)$.

$dp(i,j)$	R	...	2	1
1				
2				
...				
C				

7. Time complexity

There are $R * C$ subproblems and each subproblems is $O(1)$ hence the overall time complexity of the algorithm is $O(RC)$.